Deep Reinforcement Learning from Human Preferences
(Christiano et. al., 2017)

Jessica Yung

15 November 2017
What is 'learning from human preferences'?

Agent receives higher reward for choosing the (sequence of) action(s) that the human prefers
Outline

1. Problem
2. Method
3. Results
4. Discussion Points

Paper: Deep Reinforcement Learning from Human Preferences (Christiano et. al., 2017)
Problem 1: No well-specified reward function
Problem 1: No well-specified reward function
Problem 2: Reward function is not helpful for learning
Overview: Usual setup

```
ENV \rightarrow \text{Policy } \pi \rightarrow \text{Trajectories}
```

max r
Overview: Paper setup

Diagram:

- Environment (ENV) inputs
- Policy $\pi$ maximizes $\sum r_t$
- Trajectories
- Queries when $r$ uncertain
- Optimize parameters of $\hat{r}$ based on feedback
- Human feedback

Legend:
- $\pi$: Policy
- $r_t$: Reward at time $t$
- $\hat{r}$: Estimated reward
Three components of method

1. RL algo maximises sum of predicted rewards
2. Human gives feedback
3. Classifier learns reward function from human feedback
1. RL Algo

- Used algos that have been found to work well in each environment
  - Atari: Advantage actor-critic (A2C; Mnih et al., 2016)
  - Robotics: Trust region policy optimisation (TRPO; Schuman et al., 2015)
- Made minor changes
2. Eliciting Human Preferences

Source: RLTeacher, Tom Brown’s implementation of the paper
3. Fitting the Reward Function: Reward Function

- Reward function generates preferences over *trajectory segments*
- Assume human’s probability of preferring a segment depends exponentially on the value of the latent reward summed over the length of the clip

\[ \hat{P}[\sigma^1 \succ \sigma^2] = \frac{\exp \sum \hat{r}(o^1_t, a^1_t)}{\exp \sum \hat{r}(o^1_t, a^1_t) + \exp \sum \hat{r}(o^2_t, a^2_t)} \]
Choose \( \hat{r} \) to minimise the cross-entropy loss between predictions and human labels

\[
\text{loss}(\hat{r}) = - \sum_{(\sigma_1, \sigma_2, \mu) \in D} \mu(1) \log \hat{P}[\sigma_1 > \sigma_2] + \mu(2) \log \hat{P}[\sigma_2 > \sigma_1]
\]

Difference in predicted reward of two trajectory segments estimates the probability one is chosen over the other by the human.
- Similar to Elo in chess etc.
3. Fitting the Reward Function: Subleties

Changes shown to be helpful in experiments:

- Use an ensemble of predictors for the reward function
- Regularisation:
  - hold out $1/\epsilon$ of the data for validation, use $l_2$ regularisation, apply dropout in some domains
- Assume 10% chance the human responds uniformly at random
Experiment setup: Environments

- Three environments:
  - Simulated robotics
  - Atari arcade games
  - Novel behaviours

- Small changes to environment to hide explicit rewards

- Implemented using TensorFlow in OpenAI Gym
Experiment setup: Reward functions

1. Baseline: RL training using real reward
2. Synthetic feedback
   - Preferences reflect true reward
3. Human feedback
Simulated Robotics Demo
1. Simulated Robotics: Results

![Graphs showing simulation results for different agents including walker, hopper, swimmer, cheetah, ant, reacher, double-pendulum, and pendulum.]
1. Simulated Robotics: Results

- Does better than real reward for ant, cheetah
- Does reasonably well compared to real reward in other tasks
- Synthetic feedback (1400 queries) often gives better performance than with real reward
2. Atari

Arcade games with limited actions.
2. Atari: Results

![Graphs showing Atari game results](image)

- **beamrider**
- **breakout**
- **pong**
- **qbert**

**Axes:**
- X-axis: Timestep (up to 1e7)
- Y-axis: Reward

**Legend:**
- RL
- 10k synthetic labels
- 5.6k synthetic labels
- 3.3k synthetic labels
- 5.5k human labels
2. Atari: Results

- Does poorly in Breakout, Qbert
  - Less clear how good short clips are
- Does better than using real reward in Enduro
  - Real reward is not a clear indicator of performance
  - Meaningful rewards are sparse
3. Novel Behaviours

Human feedback training process, Results (videos)
Discussion Points

- Poor performance of offline reward predictor training
  - Reward or env distributions may be nonstationary
- Trajectory segment length
- Comparisons vs scores
Appendix 1: Fitting the Reward Function

- Interpret reward function estimate $\hat{r}$ as a preference-predictor if we view $\hat{r}$ as a latent factor explaining the human’s judgements.

- Assume human’s probability of preferring a segment $\sigma^i$ depends exponentially on the value of the latent reward summed over the length of the clip:

$$\hat{P}[\sigma^1 > \sigma^2] = \frac{\exp \sum \hat{r}(o^1_t,a^1_t)}{\exp \sum \hat{r}(o^1_t,a^1_t) + \exp \sum \hat{r}(o^2_t,a^2_t)}$$
Appendix 1.2: Fitting the Reward Function

Choose $\hat{r}$ to minimise the cross-entropy loss between predictions and human labels:

$$\text{loss}(\hat{r}) = - \sum_{(\sigma_1, \sigma_2, \mu) \in D} \mu(1) \log \hat{P}[\sigma_1 > \sigma_2] + \mu(2) \log \hat{P}[\sigma_2 > \sigma_1]$$